APPENDIX A

Media Object Server (MOS™) Protocol v2.6

MOS Protocol version 2.6
Document Revision WD-2001-08-09

Revision Date Thursday, August 09, 2001

Copyright Notice

License

1. You can use and implement all or some of the messages defined by the protocol.
2. You may not modify message names, order of defined tags within a message, tag structure, defined values, standards and constants.
3. You may not process messages in a means that is dependent on non-standard messages, tags or structures.
4. You may add additional tags to messages, but the modified messages must contain the defined minimum required tags for each message, in the order defined by this document.
5. Implementations of the MOS Protocol following the above guidelines may claim to be "MOS™ Compatible" or "MOS™ v2.6 Compatible"
6. If You add additional tags, it is strongly recommended these be included and encapsulated only within the provided <mosExternalMetadata> structure and not inserted into the pre-defined body of the message.
7. It is inappropriate to make representations of MOS Compatibility unless you have followed these guidelines.

Abstract

MOS is a three year old, evolving protocol for communications between Newsroom Computer Systems (NCS) and Media Object Servers (MOS) such as Video Servers, Audio Servers, Still Stores, and Character Generators. The MOS Protocol development is supported through cooperative collaboration among equipment vendors, software vendors and end users.

Status of this document

This document reflects changes to the MOS protocol made during the MOS meeting at NAB 2001 and is referred to as MOS v2.6. Logic and message content of MOS v 2.5 have undergone minor changes based on agreements reached in the NAB 2001 meeting. Three additional messages and one additional data structure have been added to v2.6 and are noted below in **bold red**.

How to use this document

The document contains bookmarks and hyperlinks. The Table of Contents and other areas contain underlined. Depending on the viewer application used, clicking or double clicking on underlined links should take the viewer to the relevant portion of the document.

Examples of each MOS message and data structure are included. These messages may be used for testing. Developers are encouraged to cut these messages from the document, change the value of ID tags as appropriate, and paste the modified messages into validation tools. Other than these example messages, validation tools are not provided by the MOS Group.

Media Object Server Protocol 2.6

Table of Contents

1. Introduction

This document reflects changes to the MOS protocol made during the MOS meeting at NAB 2001.

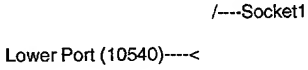
Three new messages and one new data structure were added: mosItemReplace, roMetadataReplace, roStoryMove, and mosExternalMetadata.

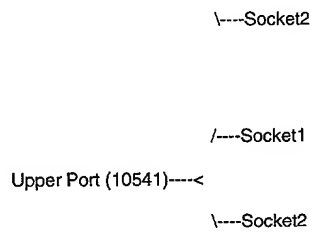
A reminder: MOS equipment should ignore, without error, any unknown tags in a message, so long as the message or structure contains properly formatted XML content and the minimum subset of required MOS tags for that message or structure.

1.1 Message Exchange

Both the NCS and MOS can originate messages. Transmitted messages require a response from the receiver before the transmitter will attempt to send the next message in the queue belonging to that specific port (either upper or lower). Upper and lower port messages are not related so that while a machine is waiting for a response on the lower port it may continue to have a dialog on the upper port.

Note: "Two Ports - Four Sockets" Each pair of communicating machines uses two ports. Each machine must be able to accept and handle messages on a minimum of two sockets per port. Once established, socket connections do not need to be dropped and then re-established between messages. Generally, the acknowledgment of a message should be sent down the same socket on which the original message was received. However, machines should be able to handle situations in which each message arrives in a separate, discrete socket session (though this is not very efficient).





Note: "Multiple MOS Connections" Each machine (NCS and MOS) should be capable of establishing and maintaining connections to multiple systems of the opposite type. e.g. An NCS should be capable of connecting to multiple Media Object Servers. Media Object Servers should also be capable of connecting to multiple NCSs.

1.2 Identification

In practice the MOS and NCS character names are predefined in each system and IP addresses associated with each name.

1.3 Encoding

The supported character encoding is ISO 10646 (Unicode) in UCS-2, as defined in The Unicode Standard, version 2.0. All MOS message contents are transmitted in Unicode, high-order byte first.

2. MOS Message Format

The MOS Protocol is fundamentally a tagged text data stream. In the version 2.x, data fields are character delimited using Extensible Markup Language (XML™) tags defined in the MOS Data Type Definition (DTD). In MOS v1.x data fields were delimited using a proprietary format.

2.1 Extensible Markup Language (XML)

The syntax of MOS v2.6 is an application of XML, an international standard for describing document content structure. XML is a simple, flexible text format based on SGML (ISO 8879). XML is an abbreviated version of SGML, to make it easier for you to define your own document types, and to make it easier for programmers to write programs to handle them. It omits the more complex and less-used parts of SGML in return for the benefits of being easier to write applications, easier to understand, and more suited to delivery and interoperability over the Web.

All tags are case sensitive. All MOS messages must be well formed XML, but are not required to be valid.

Each MOS message begins with the root tag ("mos"), followed by the MOS and NCS ID ("mosID" and "ncsID"), and followed by the message type. Data follows in tagged form.

Vendors are encouraged to add CR/LF and Tabs within a message to improve readability for debugging purposes.

2.2 Message Acknowledgement

When a message is sent by one device, it will not send another message until it receives an acknowledgement ("ACK") or error ("NACK") from the other device. Vendors should reset and/or retry when a timeout occurs.

2.3 Message Transport

MOS Lower Port (10540) is defined as the default TCP/IP port on which the NCS will accept connections from MOS devices. Multiple simultaneous connections are supported. This socket is referred to as "Media Object Metadata" port in the Message Types section.

MOS Upper Port (10541) is defined as the default TCP/IP port on which the MOS will accept connections from the NCS. Multiple simultaneous connections are supported. This socket is referred to as "Running Order" port in the Message Types section.

Ports 10520 and 10521 were specified as Lower and Upper Ports in previous versions of the MOS Protocol. Beginning in version 2.5 these ports are vendor selectable but site specific. All MOS enabled machines within a site or enterprise should communicate using the same ports.

Because some vendors reported problems using port 10521 with Microsoft Windows NT the new port numbers used as examples are now 10540 and 10541.

For example, an NCS initiated create running order command and the MOS' associated ACK would take place on MOS Upper Port (10541). Object updates sent from the MOS and the associated NCS ACK would take place on MOS Lower Port (10540).

2.4 Unknown Tags

Should a MOS or NCS encounter an unknown message or data tag the device should ignore the tag and the data and continue to process the message. Unknown data tags should not generate an application error. The application has the option of indicating a warning.

2.5 Object Description Format

Object description is restricted to plain Unicode UCS-2 text that includes Tab, CR, LF and the optional markup for paragraphs, tabs and emphasis. Formatted text such as HTML, RTF, etc. will not be allowed in the unstructured description area.

2.6 Languages

Data tags and constants are formatted as English.

The only Data Fields that can contain other languages are:

createdBy
modifiedBy
roSlug
storySlug

storyBody
itemSlug
description

And the data structure mosExternalMetadata

2.7 Numbers

Numbers are formatted as their text equivalent, e.g.:
The decimal number 100 is represented as text "100".
Hex FF55 is represented as text "0xFF55" or "xFF55".

2.8 Running Orders

1) Running Order (Unique ID - may appear only once in the NCS)
2) Story (Unique ID - may appear only once in the RO)
3) Item (Unique ID - may appear only once in a story)
4) Object (Unique ID - may appear only once in an item)

It is assumed that all Unique ID's (UID's) created by one machine are respected by others.

Order of data fields within an item is significant.

Items are sent in the order they should be played.

Order of items is significant.

Multiple Running Orders may contain the same Story.

Running Orders may contain zero or more Stories.

Multiple stories can contain the same Object referenced by different Items.

Stories can contain multiple Items.

Item IDs may appear only once in a Story, but can appear in other Stories.

Objects can appear multiple times in a Story, but only one object may appear in an Item.

A Running Order Item is defined by the combination Running Order.Story.Item and contains the UID's of the Running Order, Story and Item which together can identify a unique Item within a Running Order. Additions, deletions, and moves within the running order are referenced in this way to the Item.

2.9 Samples

Still Store and Character Generator media objects are defined as having 1 sample per second. They are special cases that require the NCS and MOS applications to understand they do not change every second.

MOS Messages

In the Structural Outline sections below use of "?" specifies optional element, "+" specifies one or more of the element, and "*"specifies zero or more of the element. Elements without any of these three special characters are required.

Tags enclosed in parenthesis and separated by "|" represent a selection.

The Syntax section shows the definition of non-terminals for this message from the DTD.

Examples shown are representative of syntax only and do not represent samples from any system.

3. Object Messages

3.1 mosAck - Acknowledge MOS Object Description

Purpose

The MOS Acknowledgement for the MOS OBJ message.

Response

N/A

Port

MOS Lower Port (10540) - Media Object Metadata

Structural Outline

mos
- mosID
- ncsID
- mosAck
  - objID
  - objRev
  - status
  - statusDescription

Syntax

```
<!ELEMENT mosAck (objID, objRev, status, statusDescription)>
```

Example

```
<mos>
  <mosID>aircache.newscenter.com</mosID>
  <ncsID>ncs.newscenter.com</ncsID>
  <mosAck>
    <objID>M000123</objID>
    <objRev>1</objRev>
    <status>ACK</status>
    <statusDescription></statusDescription>
  </mosAck>
</mos>
```

3.2 mosObj - MOS Object Description

Purpose

Contains information that describes a unique MOS Object to the NCS. The NCS uses this information to search for and reference the MOS Object.

<objGroup> tag

No specific values for this element are officially defined. Definition of values is left to the configuration and agreement of MOS connected equipment. The intended use is for site configuration of a limited number of locally named storage folders in the NCS or MOS, such as "ControlA", "ControlB" , "Raw", "Finished", etc. For editorially descriptive "category" information, it is suggested that the <externalMetadata> block be used.

ExternalMetadata Block

This data block can appear in several messages as a mechanism for transporting additional metadata, independent of schema or DTD. When found within the mosObj message, this block carries data defined external to the MOS Protocol.

External Metadata <scope> tag

The value of the <scope> tag implies through what production processes this information should travel.

A scope of "object" implies this information is generally descriptive of the object and appropriate for queries. The metadata should not be forwarded into Stories or Playlists.

A scope of "story" suggests this information may determine how the Object is to be applied in a Story. For instance, Intellectual Property Management. This information should be forwarded with the contents of a Story.

A scope of "playlist" suggests this information is specific to describing how the Object is to be published, rendered, or played to air and thus, should be included in the roCreate Play List Construction and roStorySend messages.

Scope allows systems to, optionally, roughly filter external metadata and selectively apply it to different production processes and outputs. Specifically, it is neither advisable nor efficient to send large amounts of inappropriate metadata to the Playlist in roCreate messages. In addition to these blocks of data being potentially very large, the media Object Server is, presumably, already aware of this data.

Response

mosAck

Port

MOS Lower Port (10540) - Media Object Metadata

Structural Outline

mos
- mosID
- ncsID
- mosObj
  - objID
  - objSlug
  - mosAbstract?
  - objGroup?
  - objType
  - objTB
  - objRev
  - objDur
  - status
  - objAir
  - createdBy

created
changedBy
changed
description
(p | em | tab)*

mosExternalMetadata*
mosScope?

mosSchema

mosPayload

Syntax

```
<!ELEMENT mosObj (objID, objSlug, mosAbstract, objGroup?, objType, objTB, objRev, objDur,
status, objAir, createdBy, created, changedBy, changed, description,
mosExternalMetadata*)>
<!ELEMENT description (#PCDATA | p | em | tab)*>
<!ELEMENT p (#PCDATA | em | tab)*>

<!ELEMENT mosExternalMetadata (mosScope?, mosSchema, mosPayload)>

<!ELEMENT mosScope (object | story | playlist)>

<!ELEMENT mosSchema (#PCDATA)>

<!ELEMENT mosPayload (#PCDATA)>
```

Example

```
<mos>
  <mosID>aircache.newscenter.com</mosID>
  <ncsID>ncs.newscenter.com</ncsID>
  <mosObj>
     <objID>M000123</objID>
     <objSlug>Hotel Fire</objSlug>
     <mosAbstract>
        <b>Hotel Fire</b>
        <em>vo</em>
         :30
     </mosAbstract>
     <objGroup>Show 7</objGroup>
     <objType>VIDEO</objType>
     <objTB>60</objTB>
     <objRev>1</objRev>
     <objDur>1800</objDur>
```

```
      <status>NEW</status>
      <objAir>READY</objAir>
      <createdBy>Chris</createdBy>
      <created>1998-10-31T23:39:12</created>
      <changedBy>Chris</changedBy>
      <changed>1998-10-31T23:39:12</changed>
      <description>
         <p>
            Exterior footage of
            <em>Baley Park Hotel</em>
             on fire with natural sound. Trucks are visible for the first portion of the clip.
            <em>CG locator at 0:04 and duration 0:05, Baley Park Hotel.</em>
         </p>
         <p>
            <tab/>
            Cuts to view of fire personnel exiting hotel lobby and cleaning up after the fire is
out.
         </p>
         <p>
            <em>Clip has been doubled for pad on voice over.</em>
         </p>
      </description>
      <mosExternalMetadata>
         <mosScope>STORY</mosScope>
         <mosSchema>http://MOSA4.com/mos/supported_schemas/MOSAXML2.08</mosSchema>
         <mosPayload>
            <Owner>SHOLMES</Owner>
            <ModTime>20010308142001</ModTime>
            <mediaTime>0</mediaTime>
            <TextTime>278</TextTime>
            <ModBy>LJOHNSTON</ModBy>
            <Approved>0</Approved>
            <Creator>SHOLMES</Creator>
         </mosPayload>
      </mosExternalMetadata>
   </mosObj>
</mos>
```

3.3 mosReqObj - Request Object Description

Purpose

Message used by NCS to request object description.

Response

mosObj - if objID is found
mosAck - otherwise

Port

MOS Lower Port (10540) - Media Object Metadata

Structural Outline

mos
  mosID
  ncsID
  mosReqObj
    objID

Syntax

```
<!ELEMENT mosReqObj (objID)>
```

Example

```
<mos>
   <mosID>aircache.newscenter.com</mosID>
   <ncsID>ncs.newscenter.com</ncsID>
   <mosReqObj>
      <objID>M000123</objID>
   </mosReqObj>
</mos>
```

3.10 Object Resynchronization/Rediscovery

3.11 mosReqAll - Request All Object Data from MOS

Purpose

Method for the NCS to request the MOS send it mosObj messages for every Object in the MOS.

Pause, when greater than zero, indicates the number of seconds to pause between MOS objects in the mosListAll message. Pause of zero indicates that all objects should be sent using mosObj messages.

Response

mosListAll - if pause > 0
mosObj - otherwise

Port

MOS Lower Port (10540) - Media Object Metadata

Structural Outline

mos
  mosID
  ncsID
  mosReqAll
    pause

Syntax

```
<!ELEMENT mosReqAll (pause)>
```

Example

```
<mos>
  <mosID>aircache.newscenter.com</mosID>
  <ncsID>ncs.newscenter.com</ncsID>
  <mosReqAll>
     <pause>0</pause>
  </mosReqAll>
</mos>
```

3.12 mosListAll - Listing of All Object Data from MOS

Purpose

Send MOS object descriptions in a format similar to mosObj sequentially from the MOS to the NCS in response to the mosReqAll message. There is a pause between each object as defined in the mosReqAll message.

Response

None

Port

MOS Lower Port (10540) - Media Object Metadata

Structural Outline

mos
  mosID
  ncsID
  mosListAll
    (objID, objSlug, mosAbstract?, objGroup?, objType, objTB, objRev, objDur, status, objAir, createdBy, created, changedBy, changed, description, mosExternalMetadata*)+

Syntax

```
<!ELEMENT mosListAll ((objID, objSlug, mosAbstract?, objGroup?, objType, objTB, objRev,
objDur, status, objAir, createdBy, created, changedBy, changed, description,
mosExternalMetadata*)*)>
<!ELEMENT description (#PCDATA | p | em | tab)*>
<!ELEMENT p (#PCDATA | em | tab)*>
```

Example

```
<mos>
  <mosID>aircache.newscenter.com</mosID>
  <ncsID>ncs.newscenter.com</ncsID>
  <mosListAll>
    <objID>M000123</objID>
    <objSlug>HOTEL FIRE</objSlug>
    <objType>VIDEO</objType>
    <objCategory>RAW</objGroup>
    <objTB>60</objTB>
    <objRev>3</objRev>
    <objDur>1530</objDur>
    <status>UPDATED</status>
    <objAir>READY</objAir>
    <createdBy>Chris</createdBy>
    <created>1998-10-31T23:39:12</created>
    <changedBy>Chris</changedBy>
    <changed>1998-11-01T14:35:55</changed>
    <description>
```

Exterior footage of

Baley Park Hotel

on fire with natural sound. Trucks are visible for the first portion of the clip.

CG locator at 0:04 and duration 0:05, Baley Park Hotel.

<tab/>

Cuts to view of fire personnel exiting hotel lobby and cleaning up after the fire is out.

Clip has been doubled for pad on voice over.

</description>

<objID>M000224</objID>

<objSlug>COLSTAT MURDER:VO</objSlug>

<objType>VIDEO</objType>

<objTB>60</objTB>

<objRev>4</objRev>

<objDur>800</objDur>

<status>UPDATED</status>

<objAir>READY</objAir>

<createdBy>Phil</createdBy>

<created>1998-11-01T15:19:01</created>

<changedBy>Chris</changedBy>

<changed>1998-11-01T15:21:15</changed>

<description>VOICE OVER MATERIAL OF COLSTAT MURDER SITES SHOT ON 1-NOV.</description>

```
        <mosExternalMetadata>
            <mosScope>STORY</mosScope>
            <mosSchema>http://MOSA4.com/mos/supported_schemas/MOSAXML2.08</mosSchema>
            <mosPayload>
                <Owner>SHOLMES</Owner>
                <ModTime>20010308142001</ModTime>
                <mediaTime>0</mediaTime>
                <TextTime>278</TextTime>
                <ModBy>LJOHNSTON</ModBy>
                <Approved>0</Approved>
                <Creator>SHOLMES</Creator>
            </mosPayload>
        </mosExternalMetadata>
    </mosListAll>
</mos>
```

3.20 Object and Item management

3.21 mosObjCreate – MOS Object Create

Purpose

Allows an NCS to request the Media Object Server to create a Media Object with specific metadata associated with it.

Response

mosAck (if object can be created then status description = objID)
NACK (if object CANNOT be created. status description = reason for error)
mosObj

Port

MOS Lower Port (10540) – MOS Object

Structural Outline

mos
mosID
ncsID
mosObjCreate
objSlug

objGroup?
objType
objTB
objDur?
time?
createdBy?
description?

mosExternalMetadata*

Syntax

```
<!ELEMENT mosObjCreate (objSlug, objGroup?, objType, objTB, objDur?, time?, createdBy?,
description?, mosExternalMetadata*)>
<!ELEMENT description (#PCDATA | p | em | tab)*>

<!ELEMENT p (#PCDATA | em | tab)*>
```

Example

```
<mos>
  <mosID>videoserver.station.com</mosID>
  <ncsID>ncs.station.com</ncsID>
  <mosObjCreate>
     <objSlug>Hotel Fire</objSlug>
     <objGroup>Show 7</objGroup>
     <objType>VIDEO</objType>
     <objTB>60</objTB>
     <objDur>1800</objDur>
     <createdBy>Chris</createdBy>
     <description>
        <p>
           Exterior footage of
           <em>Baley Park Hotel</em>
            on fire with natural sound. Trucks are
       visible for the first portion of the clip.
           <em>CG locator at 0:04 and duration 0:05, Baley Park
       Hotel.</em>
        </p>
        <p>
           <tab/>
```

```
            Cuts to view of fire personnel exiting hotel lobby and cleaning up after the
fire is out.
         </p>
         <p>
            <em>Clip has been doubled for pad on voice over.</em>
         </p>
      </description>
      <mosExternalMetadata>
         <mosScope>STORY</mosScope>
         <mosSchema>http://NCSA4.com/mos/supported_schemas/NCSAXML2.08</mosSchema>
         <mosPayload>
            <Owner>SHOLMES</Owner>
            <ModTime>20010308142001</ModTime>
            <mediaTime>0</mediaTime>
            <TextTime>278</TextTime>
            <ModBy>LJOHNSTON</ModBy>
            <Approved>0</Approved>
            <Creator>SHOLMES</Creator>
         </mosPayload>
      </mosExternalMetadata>
   </mosObjCreate>
</mos>
```

3.22 mosItemReplace – Replace one Item Reference with another

Purpose

This message allows a media Object Server to replace an Item Reference in a Story with new metadata values and/or additional tags. The Story must be in a MOS Active Play List. Thus, this message is in the "ro" family of messages rather than the "mos," or lower port, family. However, this message is initiated by the media Object Server, rather than the NCS.

Behavior

This message must reference as existing unique Item Reference in a MOS Active Play List through the values of ncsID, roID, storyID, and itemID.

If metadata tags in the mosItemReplace message already exist in the target Item Reference, values within the Item Reference will be replaced by the values in the mosItemReplace message.

If the metadata tags do not already exist in the target Item Reference they will be added.

If a mosExternalMetadata block is included in the mosItemReplace message, it will replace an existing mosExternalMetadata block **only** if the values of <mosSchema> in the two blocks match, and only for the first occurrence of a block with a matching <mosSchema> tag. Otherwise the mosExternalMetadata block will be added to the target Item Reference.

If the ItemID in the mosItemReplace message does not match an existing ItemID in the specified Story then no action will be taken and the mosItemReplace message will be replied to with an roAck message specifying the Item values in the mosItemReplace message and carrying a status value of "NACK."

Response

roAck

roStoryReplace

roStorySend

Subsequent messages

roStoryReplace – A successful mosItemReplace operation will result in a change to an Item reference embedded in a Story. This new information must now be placed in associated MOS Playlists. The roStoryReplace message will replace the old item reference in the playlist with the newly updated item reference from this Story.

roStorySend – A successful mosItemReplace operation will result in a change in the body of a Story. This change must be sent back out if an roStorySend target has been defined for the RO.

Port

MOS Upper Port (10541) - Running Order

Structural Outline

mos
- mosID
- ncsID
- mosItemReplace
  - roID
  - storyID
    - item*
    - itemID

itemSlug?
objID
mosID

mosAbstract?
itemChannel?
itemEdStart?
itemEdDur?
itemTrigger?
macroIn?
macroOut?

mosExternalMetadata*

Syntax

```
<!ELEMENT mosItemReplace (roID, storyID, item)>
<!ELEMENT item (itemID, itemSlug?, objID, mosID, mosAbstract?, itemChannel?,
itemEdStart?, itemEdDur?, itemTrigger?, macroIn?, macroOut?, mosExternalMetadata*>

<!ELEMENT mosExternalMetadata (mosScope?, mosSchema, mosPayload)>
```

Example

```
<mos>
   <mosID>aircache.newscenter.com</mosID>
   <ncsID>ncs.newscenter.com</ncsID>
   <mosItemReplace>
      <roID>5PM</roID>
      <storyID>HOTEL FIRE</storyID>
      <item>
         <itemID>30848</itemID>
         <objID>M000627</objID>
         <mosID>testmos.enps.com</mosID>
         <itemEdStart>0</itemEdStart>
         <itemEdDur>815</itemEdDur>
         <macroIn>c01/l04/dve07</macroIn>
         <macroOut>r00</macroOut>
         <mosExternalMetadata>
            <mosScope>PLAYLIST</mosScope>
            <mosSchema>HTTP://VVENDOR/MOS/supportedSchemas/vvend280</mosSchema>
            <mosPayload>
               <trigger>837</trigger>
```

```
                <key>110</key>
                <fade>17</fade>
                <efxTime>15</efxTime>
             </mosPayload>
          </mosExternalMetadata>
       </item>
   </mosItemReplace>
</mos>
```

4. ro (Running Order) basic messages for playlist construction

4.1 roAck - Acknowledge Running Order

Purpose

MOS response to receipt of any Running Order command.

Response

None

Port

MOS Upper Port (10541) - Running Order

Structural Outline

mos
- mosID
- ncsID
- roAck
  - roID
  - roStatus
  - (storyID, itemID, objID, status)*

Syntax

```
<!ELEMENT roAck (roID, roStatus, (storyID, itemID, objID, status)*)>
```

Example

```
<mos>
   <mosID>aircache.newscenter.com</mosID>
   <ncsID>ncs.newscenter.com</ncsID>
```

```
    <roAck>
       <roID>96857485</roID>
       <roStatus>Unknown object M000133</roStatus>
       <storyID>5983A501:0049B924:8390EF2B</storyID>
       <itemID>0</itemID>
       <objID>M000224</objID>
       <status>LOADED</status>
       <storyID>3854737F:0003A34D:983A0B28</storyID>
       <itemID>0</itemID>
       <objID>M000133</objID>
       <status>UNKNOWN</status>
    </roAck>
</mos>
```

4.2 roCreate - Create Running Order

Purpose

Message from the NCS to the MOS that defines a new Running Order.

Response

roAck

Port

MOS Upper Port (10541) - Running Order

Structural Outline

mos
- mosID
- ncsID
- roCreate
  - roID
  - roSlug
  - roChannel?
  - roEdStart?
  - roEdDur?
  - roTrigger?
  - macroIn?
  - macroOut?

mosExternalMetadata*
story*
storyID
storySlug?

storyNum?

mosExternalMetadata*
item*
itemID
itemSlug?
objID
mosID

mosAbstract?
itemChannel?
itemEdStart?
itemEdDur?
itemTrigger?
macroIn?
macroOut?

mosExternalMetadata*

Syntax

```
<!ELEMENT roCreate (roID, roSlug, roChannel?, roEdStart?, roEdDur?, roTrigger?, macroIn?,
macroOut?, mosExternalMetadata*, story*)>
<!ELEMENT story (storyID, storySlug?, storyNum?, mosExternalMetadata*,  item*)>
<!ELEMENT item (itemID, itemSlug?, objID, mosID, mosAbstract?, itemChannel?,
itemEdStart?, itemEdDur?, itemTrigger?, macroIn?, macroOut?, mosExternalMetadata*)>
```

Example

```
<mos>
   <mosID>aircache.newscenter.com</mosID>
   <ncsID>ncs.newscenter.com</ncsID>
   <roCreate>
      <roID>96857485</roID>
      <roSlug>5PM RUNDOWN</roSlug>
      <roEdStart>1999-04-17T17:02:00</roEdStart>
      <roEdDur>00:58:25</roEdDur>
      <story>
         <storyID>5983A501:0049B924:8390EF2B</storyID>
         <storySlug>COLSTAT MURDER</storySlug>
         <storyNum>A5</storyNum>
         <item>
```

```
            <itemID>0</itemID>
            <itemSlug>COLSTAT MURDER:VO</itemSlug>
            <objID>M000224</objID>
            <mosID>testmos.enps.com</mosID>
            <itemEdDur>645</itemEdDur>
            <itemTrigger>CHAINED</itemTrigger>
            <mosExternalMetadata>
               <mosScope>PLAYLIST</mosScope>
               <mosSchema>http://MOSA4.com/mos/supported_schemas/MOSAXML2.08</mosSchema>
               <mosPayload>
                  <Owner>SHOLMES</Owner>
                  <transitionMode>2</transitionMode>
                  <transitionPoint>463</transitionPoint>
                  <source>a</source>
                  <destination>b</destination>
               </mosPayload>
            </mosExternalMetadata>
         </item>
      </story>
      <story>
         <storyID>3854737F:0003A34D:983A0B28</storyID>
         <storySlug>AIRLINE INSPECTIONS</storySlug>
         <storyNum>A6</storyNum>
         <item>
            <itemID>0</itemID>
            <objID>M000133</objID>
            <mosID>testmos.enps.com</mosID>
            <itemEdStart>55</itemEdStart>
            <itemEdDur>310</itemEdDur>
            <mosExternalMetadata>
               <mosScope>PLAYLIST</mosScope>
               <mosSchema>http://MOSA4.com/mos/supported_schemas/MOSAXML2.08</mosSchema>
               <mosPayload>
                  <Owner>SHOLMES</Owner>
```

```
                    <transitionMode>2</transitionMode>
                    <transitionPoint>463</transitionPoint>
                    <source>a</source>
                    <destination>b</destination>
                </mosPayload>
            </mosExternalMetadata>
        </item>
    </story>
  </roCreate>
</mos>
```

4.3 roReplace - Replace Running Order

Purpose

Replaces an existing Running Order definition in the MOS with another one sent from the NCS.

Response

roAck

Port

MOS Upper Port (10541) - Running Order

Structural Outline

mos
- mosID
- ncsID
- roReplace
  - roID
  - roSlug
  - roChannel?
  - roEdStart?
  - roEdDur?
  - roTrigger?
  - macroIn?
  - macroOut?
  - mosExternalMetadata*
  - story*
    - storyID

storySlug?

storyNum?

mosExternalMetadata*
item*
itemID
itemSlug?
objID
mosID

mosAbstract?
itemChannel?
itemEdStart?
itemEdDur?
itemTrigger?
macroIn?
macroOut?

mosExternalMetadata*

Syntax

```
<!ELEMENT roReplace (roID, roSlug, roChannel?, roEdStart?, roEdDur?, roTrigger?,
macroIn?, macroOut?, mosExternalMetadata*, story*)>
<!ELEMENT story (storyID, storySlug?, storyNum?, mosExternalMetadata*,  item*)>
<!ELEMENT item (itemID, itemSlug?, objID, mosID, mosAbstract?, itemChannel?,
itemEdStart?, itemEdDur?, itemTrigger?, macroIn?, macroOut?, mosExternalMetadata*)>
```

Example

```
<mos>
  <mosID>aircache.newscenter.com</mosID>
  <ncsID>ncs.newscenter.com</ncsID>
  <roReplace>
    <roID>96857485</roID>
    <roSlug>5PM RUNDOWN</roSlug>
    <story>
      <storyID>5983A501:0049B924:8390EF2B</storyID>
      <storySlug>COLSTAT MURDER</storySlug>
      <storyNum>A1</storyNum>
      <item>
        <itemID>0</itemID>
        <itemSlug>COLSTAT MURDER:VO</itemSlug>
        <objID>M000224</objID>
        <mosID>testmos.enps.com</mosID>
```

```
            <itemEdDur>645</itemEdDur>
            <itemTrigger>CHAINED</itemTrigger>
            <mosExternalMetadata>
               <mosScope>PLAYLIST</mosScope>
               <mosSchema>http://MOSA4.com/mos/supported_schemas/MOSAXML2.08</mosSchema>
               <mosPayload>
                  <Owner>SHOLMES</Owner>
                  <transitionMode>2</transitionMode>
                  <transitionPoint>463</transitionPoint>
                  <source>a</source>
                  <destination>b</destination>
               </mosPayload>
            </mosExternalMetadata>
         </item>
      </story>
      <story>
         <storyID>3852737F:0013A64D:923A0B28</storyID>
         <storySlug>AIRLINE SAFETY</storySlug>
         <storyNum>A2</storyNum>
         <item>
            <itemID>0</itemID>
            <objID>M000295</objID>
            <mosID>testmos.enps.com</mosID>
            <itemEdStart>500</itemEdStart>
            <itemEdDur>600</itemEdDur>
            <mosExternalMetadata>
               <mosScope>PLAYLIST</mosScope>
               <mosSchema>http://MOSA4.com/mos/supported_schemas/MOSAXML2.08</mosSchema>
               <mosPayload>
                  <Owner>SHOLMES</Owner>
                  <transitionMode>2</transitionMode>
                  <transitionPoint>463</transitionPoint>
                  <source>a</source>
                  <destination>b</destination>
```

```
            </mosPayload>
          </mosExternalMetadata>
        </item>
      </story>
    </roReplace>
</mos>
```

4.4 roMetadataReplace – Replace RO metadata without deleting the RO structure

Purpose

This message allows metadata associated with a running order to be replaced without deleting the running order and sending the entire running order again.

Behavior

This message must reference an existing running order

If metadata tags in the roMetadataReplace message already exist in the target RO, values within the RO will be replaced by the values in the roMetadataReplace message.

If the metadata tags do not already exist in the target RO they will be added.

If a mosExternalMetadata block is included in the roMetadataReplace message, it will replace an existing mosExternalMetadata block **only** if the values of mosSchema in the two blocks match. Otherwise the mosExternalMetadata block will be added to the target RO.

If the ROID in the roMetadataReplace message does not match an existing ROID then no action will be taken and the roMetadataReplace message will be replied to with an roAck message which carrying a status value of "NACK."

Response

roAck

Port

MOS Upper Port (10541) - Running Order

Structural Outline

mos
- mosID
- ncsID
- roMetadataReplace
  - roID
  - roSlug
  - roChannel?
  - roEdStart?
  - roEdDur?
  - roTrigger?
  - roMacroIn?
  - roMacroOut?
  - mosExternalMetadata?

Syntax

```
<!ELEMENT roMetadataReplace (roID, roSlug, roChannel?, roEdStart?, roEdDur?, roTrigger?,
roMacroIn?, roMacroOut?, mosExternalMetadata?)>
```

Example

```
<mos>
   <mosID>aircache.newscenter.com</mosID>
   <ncsID>ncs.newscenter.com</ncsID>
   <roMetadataReplace>
      <roID>96857485</roID>
      <roSlug>5PM RUNDOWN</roSlug>
      <roEdStart>1999-04-17T17:02:00</roEdStart>
      <roEdDur>00:58:25</roEdDur>
      <mosExternalMetadata>
         <mosScope>PLAYLIST</mosScope>
         <mosSchema>http://MOSA4.com/mos/supported_schemas/MOSAXML2.08</mosSchema>
         <mosPayload>
            <Owner>SHOLMES</Owner>
            <transitionMode>2</transitionMode>
            <transitionPoint>463</transitionPoint>
            <source>a</source>
```

```
            <destination>b</destination>
         </mosPayload>
      </mosExternalMetadata>
   </roMetadataReplace>
</mos>
```

4.5 roDelete - Delete Running Order

Purpose

Deletes a Running order in the MOS.

Response

roAck

Port

MOS Upper Port (10541) - Running Order

Structural Outline

mos
- mosID
- ncsID
- roDelete
  - roID

Syntax

```
<!ELEMENT roDelete (roID)>
```

Example

```
<mos>
   <mosID>aircache.newscenter.com</mosID>
   <ncsID>ncs.newscenter.com</ncsID>
   <roDelete>
      <roID>49478285</roID>
   </roDelete>
</mos>
```

4.10 ro Synchronization, Discovery and Status

4.11 roReq - Request Running Order

Purpose

Request for a complete build of a Running Order Playlist.

NOTE: This message can be used by either NCS or MOS

A MOS can use this to "resync" it's Playlist with the NCS Running Order or to obtain a full description of the Playlist at any time.

An NCS can use this as a diagnostic tool to check the order of the Playlist constructed in the MOS versus the sequence of Items in the Running Order.

Response

roList or roAck (roAck with a NACK value is sent if the Running Order ID is not valid or roList cannot be returned for some reason)

Port

MOS Upper Port (10541) - Running Order

Structural Outline

mos
- mosID
- ncsID
- roReq
  - roID

Syntax

```
<!ELEMENT roReq (roID)>
```

Example

```
<mos>
   <mosID>aircache.newscenter.com</mosID>
   <ncsID>ncs.newscenter.com</ncsID>
   <roReq>
      <roID>96857485</roID>
   </roReq>
</mos>
```

4.12 roList - List Running Order

Purpose

A complete build or rebuild of a Running Order Playlist in response to an roReq message.

NOTE: This message can be sent by either the NCS or MOS

A MOS can use this to "resync" it's Playlist with the NCS Running Order or to obtain a full description of the Playlist at any time.

An NCS can use this as a diagnostic tool to check the order of the Playlist constructed in the MOS versus the sequence of Items in the Running Order.

roList is functionally similar to roCreate.

Response

None

Port

MOS Upper Port (10541) - Running Order

Structural Outline

mos
- mosID
- ncsID
- roList
  - roID
  - roSlug
  - roChannel?
  - roEdStart?
  - roEdDur?
  - roTrigger?
  - macroIn?
  - macroOut?
  - mosExternalMetadata*
  - story*
    - storyID
    - storySlug?
    - storyNum?
    - mosExternalMetadata*
    - item*

itemID
itemSlug?
objID
mosID

mosAbstract?
itemChannel?
itemEdStart?
itemEdDur?
itemTrigger?
macroIn?
macroOut?

mosExternalMetadata*

Syntax

```
<!ELEMENT roList (roID, roSlug, roChannel?, roEdStart?, roEdDur?, roTrigger?, macroIn?,
macroOut?, mosExternalMetadata*, story*)>
<!ELEMENT story (storyID, storySlug?, storyNum?, mosExternalMetadata*,  item*)>
<!ELEMENT item (itemID, itemSlug?, objID, mosID, mosAbstract?, itemChannel?,
itemEdStart?, itemEdDur?, itemTrigger?, macroIn?, macroOut?, mosExternalMetadata*)>
```

Example

```
<mos>
  <mosID>aircache.newscenter.com</mosID>
  <ncsID>ncs.newscenter.com</ncsID>
  <roList>
    <roID>96857485</roID>
    <roSlug>5PM RUNDOWN</roSlug>
    <story>
      <storyID>5983A501:0049B924:8390EF2B</storyID>
      <storySlug>Colstat Murder</storySlug>
      <storyNum>B10</storyNum>
      <item>
        <itemID>0</itemID>
        <itemSlug>COLSTAT MURDER:VO</itemSlug>
        <objID>M000224</objID>
        <mosID>testmos.enps.com</mosID>
        <itemEdDur>645</itemEdDur>
        <itemTrigger>CHAINED</itemTrigger>
        <mosExternalMetadata>
          <mosScope>PLAYLIST</mosScope>
```

```
            <mosSchema>http://MOSA4.com/mos/supported_schemas/MOSAXML2.08</mosSchema>
            <mosPayload>
               <Owner>SHOLMES</Owner>
               <transitionMode>2</transitionMode>
               <transitionPoint>463</transitionPoint>
               <source>a</source>
               <destination>b</destination>
            </mosPayload>
         </mosExternalMetadata>
      </item>
   </story>
   <story>
      <storyID>3854737F:0003A34D:983A0B28</storyID>
      <storySlug>Test MOS</storySlug>
      <storyNum>B11</storyNum>
      <item>
         <itemID>0</itemID>
         <objID>M000133</objID>
         <mosID>testmos.enps.com</mosID>
         <itemEdStart>55</itemEdStart>
         <itemEdDur>310</itemEdDur>
         <mosExternalMetadata>
            <mosScope>PLAYLIST</mosScope>
            <mosSchema>http://MOSA4.com/mos/supported_schemas/MOSAXML2.08</mosSchema>
            <mosPayload>
               <Owner>SHOLMES</Owner>
               <transitionMode>2</transitionMode>
               <transitionPoint>463</transitionPoint>
               <source>a</source>
               <destination>b</destination>
            </mosPayload>
         </mosExternalMetadata>
      </item>
   </story>
```

```
   </roList>
</mos>
```

4.13 roReqAll - Request All Running Order Descriptions

Purpose

Request for a description of all Running Orders known by a NCS from a MOS.

Response

roListAll

Port

MOS Upper Port (10541) - Running Order

Structural Outline

- mos
  - mosID
  - ncsID
  - roReqAll

Syntax

```
<!ELEMENT roReqAll EMPTY>
```

Example

```
<mos>
   <mosID>aircache.newscenter.com</mosID>
   <ncsID>ncs.newscenter.com</ncsID>
   <roReqAll/>
</mos>
```

4.14 roListAll - List All Running Order Descriptions

Purpose

Provides a description of all Running Orders known by a NCS to a MOS.

Response

None

Port

MOS Upper Port (10541) - Running Order

Structural Outline

mos
- mosID
- ncsID
- roListAll
  - (roID, roSlug?, roChannel?, roEdStart?, roEdDur?, roTrigger?, mosExternalMetadata*)*

Syntax

```
<!ELEMENT roListAll ((roID, roSlug?, roChannel?, roEdStart?, roEdDur?, roTrigger?,
mosExternalMetadata*)*)>
```

Example

```
<mos>
  <mosID>aircache.newscenter.com</mosID>
  <ncsID>ncs.newscenter.com</ncsID>
  <roListAll>
    <roID>5PM</roID>
    <roSlug>5PM RUNDOWN</roSlug>
    <roEdDur>1733</roEdDur>
    <roEdTrigger>MANUAL</roEdTrigger>
    <mosExternalMetadata>
      <mosScope>PLAYLIST</mosScope>
      <mosSchema>http://ncsA4.com/mos/supported_schemas/NCSAXML2.08</mosSchema>
      <mosPayload>
        <Owner>SHOLMES</Owner>
        <ModTime>20010308142001</ModTime>
        <mediaTime>0</mediaTime>
        <TextTime>278</TextTime>
        <ModBy>LJOHNSTON</ModBy>
        <Approved>0</Approved>
        <Creator>SHOLMES</Creator>
      </mosPayload>
    </mosExternalMetadata>
    <mosExternalMetadata>
```

```
        <mosScope>PLAYLIST</mosScope>
        <mosSchema>http://ncsA4.com/mos/supported_schemas/NCSBBBBXML2.08</mosSchema>
        <mosPayload>
           <show>10pm</show>
           <client>network ctrl b</client>
        </mosPayload>
      </mosExternalMetadata>
   </roListAll>
</mos>
```

4.15 roStat - Status of a MOS Running Order

Purpose

Method for the MOS to update the NRC on the status of Play List. This allows the NRC to reflect the status of the Play List in the NRC Running Order Display.

Response

roAck

Port

MOS Upper Port (10541) - Running Order

Structural Outline

mos
- mosID
- ncsID
- roStat
  - roID
  - status
  - time

Syntax

```
<!ELEMENT roStat (roID, status, time)>
```

Example

```
<mos>
   <mosID>aircache.newscenter.com</mosID>
   <ncsID>ncs.newscenter.com</ncsID>
```

```
   <roStat>
      <roID>5PM</roID>
      <status>MANUAL CTRL</status>
      <time>1999-04-11T14:22:07</time>
   </roStat>
</mos>
```

4.16 roReadyToAir - Identify a Running Order as Ready to Air

Purpose

The message allows the NCS to signal the MOS that a Running Order has been editorially approved ready for air.

Response

roAck

Port

MOS Upper Port (10541) - Running Order

Structural Outline

mos
- mosID
- ncsID
- roReadyToAir
  - roID
  - roAir

Syntax

```
<!ELEMENT roReadyToAir (roID, roAir)>
```

Example

```
<mos>
   <mosID>aircache.newscenter.com</mosID>
   <ncsID>ncs.newscenter.com</ncsID>
   <roReadyToAir>
      <roID>5PM</roID>
      <roAir>READY</roAir>
   </roReadyToAir>
```

</mos>

4.2 ro Story Sequence Modification

4.21 roStoryAppend - Append Stories to Running Order

Purpose

Appends stories and all of its defined items at the end of a running order.

Response

roAck

Port

MOS Upper Port (10541) - Running Order

Structural Outline

mos
- mosID
- ncsID
- roStoryAppend
  - roID
  - story*
    - storyID
    - storySlug?
    - storyNum?
    - mosExternalMetadata*
    - item*
      - itemID
      - itemSlug?
      - objID
      - mosID
      - mosAbstract?
      - itemChannel?
      - itemEdStart?
      - itemEdDur?
      - itemTrigger?
      - macroIn?
      - macroOut?

mosExternalMetadata*

Syntax

```
<!ELEMENT roStoryAppend (roID, story+)>
<!ELEMENT story (storyID, storySlug?, storyNum?, mosExternalMetadata*, item*)>
<!ELEMENT item (itemID, itemSlug?, objID, mosID, mosAbstract?, itemChannel?,
itemEdStart?, itemEdDur?, itemTrigger?, macroIn?, macroOut?, mosExternalMetadata*)>
```

Example

```
<mos>
   <mosID>aircache.newscenter.com</mosID>
   <ncsID>ncs.newscenter.com</ncsID>
   <roStoryAppend>
      <roID>5PM</roID>
      <story>
         <storyID>V: BRIDGE COLLAPSE</storyID>
         <storySlug>Bridge Collapse</storySlug>
         <storyNum>B7</storyNum>
         <item>
            <itemID>30848</itemID>
            <objID>M000627</objID>
            <mosID>testmos.enps.com</mosID>
            <itemEdStart>0</itemEdStart>
            <itemEdDur>815</itemEdDur>
            <macroIn>c01/l04/dve07</macroIn>
            <macroOut>r00</macroOut>
            <mosExternalMetadata>
               <mosScope>PLAYLIST</mosScope>
               <mosSchema>http://MOSA4.com/mos/supported_schemas/MOSAXML2.08</mosSchema>
               <mosPayload>
                  <Owner>SHOLMES</Owner>
                  <transitionMode>2</transitionMode>
                  <transitionPoint>463</transitionPoint>
                  <source>a</source>
                  <destination>b</destination>
               </mosPayload>
```

```
            </mosExternalMetadata>
            <mosExternalMetadata>
               <mosScope>PLAYLIST</mosScope>
               <mosSchema>http://MOSA4.com/mos/supported_schemas/MOSBXML2.08</mosSchema>
               <mosPayload>
                  <rate>52</rate>
                  <background>2</background>
                  <overlay>463</overlay>
               </mosPayload>
            </mosExternalMetadata>
         </item>
         <item>
            <itemID>30849</itemID>
            <objID>M000628</objID>
            <itemEdStart>0</itemEdStart>
            <itemEdDur>815</itemEdDur>
            <macroIn>c01/l04/dve07</macroIn>
            <macroOut>r00</macroOut>
            <mosExternalMetadata>
               <mosScope>PLAYLIST</mosScope>
               <mosSchema>http://MOSA4.com/mos/supported_schemas/MOSAXML2.08</mosSchema>
               <mosPayload>
                  <Owner>SHOLMES</Owner>
                  <transitionMode>2</transitionMode>
                  <transitionPoint>463</transitionPoint>
                  <source>a</source>
                  <destination>b</destination>
               </mosPayload>
            </mosExternalMetadata>
         </item>
      </story>
   </roStoryAppend>
</mos>
```

4.22 roStoryInsert - Insert Stories in Running Order

Purpose

Inserts stories and all of its defined items before the referenced Story in the running order.

Response

roAck

Port

MOS Upper Port (10541) - Running Order

Structural Outline

mos
- mosID
- ncsID
- roStoryInsert
  - roID
  - storyID
  - story+
    - storyID
    - storySlug?
    - storyNum?
    - mosExternalMetadata*
    - item*
      - itemID
      - itemSlug?
      - objID
      - mosID
      - mosAbstract?
      - itemChannel?
      - itemEdStart?
      - itemEdDur?
      - itemTrigger?
      - macroIn?
      - macroOut?
      - mosExternalMetadata*

Syntax

```
<!ELEMENT roStoryInsert (roID, storyID, story+)>
<!ELEMENT story (storyID, storySlug?, storyNum?, mosExternalMetadata*, item*)>
<!ELEMENT item (itemID, itemSlug?, objID, mosID, mosAbstract?, itemChannel?,
itemEdStart?, itemEdDur?, itemTrigger?, macroIn?, macroOut?, mosExternalMetadata*)>
```

Example

```
<mos>
    <mosID>aircache.newscenter.com</mosID>
    <ncsID>ncs.newscenter.com</ncsID>
    <roStoryInsert>
        <roID>5PM</roID>
        <storyID>HOTEL FIRE</storyID>
        <story>
            <storyID>V: BRIDGE COLLAPSE</storyID>
            <storySlug>Bridge Collapse</storySlug>
            <storyNum>B7</storyNum>
            <item>
                <itemID>30848</itemID>
                <objID>M000627</objID>
                <mosID>testmos.enps.com</mosID>
                <itemEdStart>0</itemEdStart>
                <itemEdDur>815</itemEdDur>
                <macroIn>c01/l04/dve07</macroIn>
                <macroOut>r00</macroOut>
                <mosExternalMetadata>
                    <mosScope>PLAYLIST</mosScope>
                    <mosSchema>http://MOSA4.com/mos/supported_schemas/MOSAXML2.08</mosSchema>
                    <mosPayload>
                        <Owner>SHOLMES</Owner>
                        <transitionMode>2</transitionMode>
                        <transitionPoint>463</transitionPoint>
                        <source>a</source>
                        <destination>b</destination>
                    </mosPayload>
                </mosExternalMetadata>
                <mosExternalMetadata>
                    <mosScope>PLAYLIST</mosScope>
                    <mosSchema>http://MOSA4.com/mos/supported_schemas/MOSBXML2.08</mosSchema>
```

```
                <mosPayload>
                    <rate>52</rate>
                    <background>2</background>
                    <overlay>463</overlay>
                </mosPayload>
            </mosExternalMetadata>
        </item>
        <item>
            <itemID>30849</itemID>
            <objID>M000628</objID>
            <itemEdStart>0</itemEdStart>
            <itemEdDur>815</itemEdDur>
            <macroIn>c01/l04/dve07</macroIn>
            <macroOut>r00</macroOut>
            <mosExternalMetadata>
                <mosScope>PLAYLIST</mosScope>
                <mosSchema>http://MOSA4.com/mos/supported_schemas/MOSAXML2.08</mosSchema>
                <mosPayload>
                    <Owner>SHOLMES</Owner>
                    <transitionMode>2</transitionMode>
                    <transitionPoint>463</transitionPoint>
                    <source>a</source>
                    <destination>b</destination>
                </mosPayload>
            </mosExternalMetadata>
        </item>
    </story>
  </roStoryInsert>
</mos>
```

4.23 roStoryReplace - Replace Story with Another in a Running Order

Purpose

Replaces the referenced story with another story or stories and all of its associated Items in the

Running Order.

Response

roAck

Port

MOS Upper Port (10541) - Running Order

Structural Outline

mos
- mosID
- ncsID
- roStoryReplace
  - roID
  - storyID
  - story+
    - storyID
    - storySlug?
    - storyNum?
    - mosExternalMetadata*
    - item*
      - itemID
      - itemSlug?
      - objID
      - mosID
      - mosAbstract?
      - itemChannel?
      - itemEdStart?
      - itemEdDur?
      - itemTrigger?
      - macroIn?
      - macroOut?
      - mosExternalMetadata*

Syntax

```
<!ELEMENT roStoryReplace (roID, storyID, story+)>
<!ELEMENT story (storyID, storySlug?, storyNum?, mosExternalMetadata*,  item*)>
<!ELEMENT item (itemID, itemSlug?, objID, mosID, mosAbstract?, itemChannel?,
itemEdStart?, itemEdDur?, itemTrigger?, macroIn?, macroOut?, mosExternalMetadata*)>
```

Example

```
<mos>

   <mosID>aircache.newscenter.com</mosID>
```

```
<ncsID>ncs.newscenter.com</ncsID>
<roStoryReplace>
   <roID>5PM</roID>
   <storyID>P: PHILLIPS INTERVIEW</storyID>
   <story>
      <storyID>V: HOTEL FIRE</storyID>
      <storySlug>Hotel Fire</storySlug>
      <storyNum>C1</storyNum>
      <item>
         <itemID>30848</itemID>
         <itemSlug>Hotel Fire vo</itemSlug>
         <objID>M000702</objID>
         <mosID>testmos</mosID>
         <itemEdStart>0</itemEdStart>
         <itemEdDur>900</itemEdDur>
         <macroIn>c01/l04/dve07</macroIn>
         <macroOut>r00</macroOut>
         <mosExternalMetadata>
            <mosScope>PLAYLIST</mosScope>
            <mosSchema>http://MOSA4.com/mos/supported_schemas/MOSAXML2.08</mosSchema>
            <mosPayload>
               <Owner>SHOLMES</Owner>
               <transitionMode>2</transitionMode>
               <transitionPoint>463</transitionPoint>
               <source>a</source>
               <destination>b</destination>
            </mosPayload>
         </mosExternalMetadata>
      </item>
   </story>
   <story>
      <storyID>V: DORMITORY FIRE</storyID>
      <storySlug>Dormitory Fire</storySlug>
      <storyNum>C2</storyNum>
```

```
        <item>
            <itemID>1</itemID>
            <itemSlug>Dormitory Fire vo</itemSlug>
            <objID>M000705</objID>
            <mosID>testmos</mosID>
            <itemEdStart>0</itemEdStart>
            <itemEdDur>800</itemEdDur>
            <macroIn>c01/l04/dve07</macroIn>
            <macroOut>r00</macroOut>
            <mosExternalMetadata>
                <mosScope>PLAYLIST</mosScope>
                <mosSchema>http://MOSA4.com/mos/supported_schemas/MOSAXML2.08</mosSchema>
                <mosPayload>
                    <Owner>SHOLMES</Owner>
                    <transitionMode>2</transitionMode>
                    <transitionPoint>463</transitionPoint>
                    <source>a</source>
                    <destination>b</destination>
                </mosPayload>
            </mosExternalMetadata>
        </item>
    </story>
</roStoryReplace>
</mos>
```

4.24 roStoryMove – Move a story to a new position in the playlist

Purpose

This message allows a story to be moved to a new location in a playlist. The first storyID is the ID of the story to be moved. The second storyID is the ID of the story above which the first story is to be moved.

noteIf the second <storyID> tag is blank move to the bottom.

Response

roAck

Port

MOS Upper Port (10541) - Running Order

Structural Outline

mos
- mosID
- ncsID
- roStoryMove
  - roID
  - storyID
  - storyID

Syntax

```
<!ELEMENT roStoryMove (roID, storyID, storyID)>
```

Example

```
<mos>
  <mosID>aircache.newscenter.com</mosID>
  <ncsID>ncs.newscenter.com</ncsID>
  <roStoryMove>
     <roID>5PM</roID>
     <StoryID>V: BRIDGE COLLAPSE</StoryID>
     <StoryID>P: PHILLIPS INTERVIEW</StoryID>
  </roStoryMove>
</mos>
```

4.25 roStorySwap - Swap Positions of Stories in Running Order

Purpose

Swaps the Positions of stories and all of their associated Items in the Running Order.

Response

roAck

Port

MOS Upper Port (10541) - Running Order

Structural Outline

mos
- mosID
- ncsID
- roStorySwap
  - roID
  - storyID
  - storyID

Syntax

```
<!ELEMENT roStorySwap (roID, storyID, storyID)>
```

Example

```
<mos>
   <mosID>aircache.newscenter.com</mosID>
   <ncsID>ncs.newscenter.com</ncsID>
   <roStorySwap>
      <roID>5PM</roID>
      <storyID>V: BRIDGE COLLAPSE</storyID>
      <storyID>P: PHILLIPS INTERVIEW</storyID>
   </roStorySwap>
</mos>
```

4.26 roStoryDelete - Delete Stories from Running Order

Purpose

Deletes the referenced Stories and all associated Items from the Running Order.

Response

roAck

Port

MOS Upper Port (10541) - Running Order

Structural Outline

mos
- mosID
- ncsID
- roStoryDelete

roID
storyID+

Syntax

```
<!ELEMENT roStoryDelete (roID, storyID+)>
```

Example

```
<mos>
   <mosID>aircache.newscenter.com</mosID>
   <ncsID>ncs.newscenter.com</ncsID>
   <roStoryDelete>
      <roID>5PM</roID>
      <storyID>V: BRIDGE COLLAPSE</storyID>
      <storyID>P: PHILLIPS INTERVIEW</storyID>
   </roStoryDelete>
</mos>
```

4.30 ro Control and Status feedback

4.31 roItemStat - Status of a Single Item in a MOS Running Order

Purpose

Method for the MOS to update the NCS on the status of an individual Item in a Running Order. This allows the NCS to reflect the status of individual Items in the MOS Running Order in the NCS Running Order display.

Response

roAck

Port

MOS Upper Port (10541) - Running Order

Structural Outline

mos
  mosID
  ncsID
  roItemStat
    roID
    storyID
    itemID

objID
status
time

Syntax

```
<!ELEMENT roItemStat (roID, storyID, itemID, objID, status, time)>
```

Example

```
<mos>
   <mosID>aircache.newscenter.com</mosID>
   <ncsID>ncs.newscenter.com</ncsID>
   <roItemStat>
      <roID>5PM</roID>
      <storyID>HOTEL FIRE</storyID>
      <itemID>0</itemID>
      <objID>A0295</objID>
      <status>PLAY</status>
      <time>1999-04-11T14:13:53</time>
   </roItemStat>
</mos>
```

4.32 roItemCue – Notification of Item Event

Purpose

Allows a device, such as prompter, to send a time cue for an Item.

Description

This command allows a non MOS or NCS device to send a time cue to the parent Media Object Server (or Automation MOS) for a specific Item event. This is not a command to execute or play. Instead, this is intended to provide feedback to the parent device as to the current execution point of the program.

The values <mosID>, <roID>, <storyID>, and <itemID> are derived from the Item reference embedded in a story. The story information is assumed to be transmitted via the roStorySend message.

The Media Object Server or automation device that receives this command may use this information to update status or generate device triggers. Optionally, the message may be redirected to the NCS as a means of providing additional status information.

Response

roAck

Port

MOS Upper Port (10541) - Running Order

Structural Outline

mos
  mosID
  ncsID
  roItemCue
    mosID
    roID
    storyID
    itemID
    roEventType
    roEventTime
    mosExternalMetadata*

Syntax

```
<!ELEMENT roItemCue (mosID, roID, storyID, itemID, roEventType, roEventTime,
mosExternalMetadata*)>

<!ELEMENT roEventType (#PCDATA)>

<!ELEMENT roEventTime (#PCDATA)>
```

Example

An example of a notification message forced to the NCS:

```
<mos>

   <mosID>prompt.station.com</mosID>

   <ncsID>ncs.station.com</ncsID>

   <roItemCue>

      <mosID>videoserver.station.group.com</mosID>

      <roID>96857485</roID>

      <storyID>5983A501:0049B924:8390EF2B</storyID>

      <itemID>234343234</itemID>

      <roEventType>Prompter</roEventType>

      <roEventTime>2000-03-20T10:45:00.00</roEventTime>
```

```
      <mosExternalMetadata>
         <mosScope>PLAYLIST</mosScope>
         <mosSchema>http://ncsA4.com/mos/supported_schemas/NCSBBBBXML2.08</mosSchema>
         <mosPayload>
            <triggeredby>operator</triggeredby>
            <operatorType>prompter</operatorType>
            <netPropDelay>10</netPropDelay>
         </mosPayload>
      </mosExternalMetadata>
   </roItemCue>
</mos>
```

An example of a notification message sent to the parent MOS. Note the counterintuitive assignment of the parent MOS name to the <ncsID> field:

```
<mos>
   <mosID>prompt.station.com</mosID>
   <ncsID>videoserver.station.group.com</ncsID>
   <roItemCue>
      <mosID>videoserver.station.group.com</mosID>
      <roID>96857485</roID>
      <storyID>5983A501:0049B924:8390EF2B</storyID>
      <itemID>234343234</itemID>
      <roEventType>Prompter</roEventType>
      <roEventTime>2000-03-20T10:45:00.00</roEventTime>
      <mosExternalMetadata>
         <mosScope>PLAYLIST</mosScope>
         <mosSchema>http://ncsA4.com/mos/supported_schemas/NCSBBBBXML2.08</mosSchema>
         <mosPayload>
            <triggeredby>operator</triggeredby>
            <operatorType>prompter</operatorType>
            <netPropDelay>10</netPropDelay>
         </mosPayload>
      </mosExternalMetadata>
   </roItemCue>
</mos>
```

4.33 roCtrl – Running Order Control

Purpose

Allow basic control of a media object server via simple commands such as READY, EXECUTE, PAUSE, STOP and SIGNAL

Description

The roCtrl message allows control of a running order at three levels, the Running Order itself, Story, and Item. The commands READY, EXECUTE, PAUSE and STOP, as well as general indicator, SIGNAL, can be addressed at each level. In other words, a single command can begin EXECUTION of an entire Running Order, of a Story containing multiple Items, or of a single Item.

Response

roAck

Port

MOS Upper Port (10541) - Running Order

Structural Outline

mos
- mosID
- ncsID
- roCtrl
  - roID
  - storyID
  - itemID
  - command

  mosExternalMetadata*

Syntax

```
<!ELEMENT roCtrl (roID, storyID, itemID, command, mosExternalMetadata*)>

<!ELEMENT command (READY|EXECUTE|PAUSE|STOP|SIGNAL)>
```

Example

```
<mos>

   <mosID>aircache.newscenter.com</mosID>

   <ncsID>ncs.newscenter.com</ncsID>

   <roCtrl>
```

```
      <roID>3dedde9jd</roID>
      <storyID>A3fds3d</storyID>
      <itemID>30848</itemID>
      <command>EXECUTE</command>
   </roCtrl>
</mos>
```

4.40 ro Metadata Export

4.41 roStorySend – Send Story information, including Body of the Story

Purpose

This message enables sending the body of story from the NCS to Media Object Server. Item references (storyItem) are embedded within the story's text. These item references are not intended to be displayed on the prompter, but instead can optionally be used to send a message (roItemCue) to the media object server indicated in the embedded reference. Composed from information in the embedded item reference, the roItemCue message could be generated by the prompter as this hidden text (storyItem) scrolls past the imaginary execution/read line of the prompter display.

The storyItem information can also optionally allow the prompter vendor to display the length of the embedded object and perhaps even a countdown.

Prompters, radio systems, external archive systems, accounting systems, and potentially other systems and devices can make use of this information.

Response

roAck

Port

MOS Upper Port (10541) - Running Order

Structural Outline

mos
- mosID
- ncsID
- roStorySend
  - roID
  - storyID
  - storySlug?

storyNum?

storyBody
storyPresenter*
storyPresenterRR*
p*
em*
tab*
pi*
pkg*
b*
I*
u*
storyItem*
itemID
itemSlug?
objID
mosID

mosAbstract?
itemChannel?
itemEdStart?
itemEdDur?
itemTrigger?
macroIn?
macroOut?

mosExternalMetadta*

Syntax

```
<!ELEMENT roStorySend (roID, storyID, storySlug?, storyNum?, mosExternalMetadata*
storyBody)>

<!ELEMENT storyBody (storyPresenter*, storyPresenterRR*, p*, storyItem*)>

<!ELEMENT storyPresenter (#PCDATA)>

<!ELEMENT storyPresenterRR (#PCDATA)>

<!ELEMENT storyItem (itemID, itemSlug?, objID, mosID, mosAbstract?, itemChannel?,
itemEdStart?, itemEdDur?, itemTrigger?, macroIn?, macroOut?, mosExternalMetadata*)>

<!ELEMENT p (#PCDATA | em | tab | pi | pkg | b | i | u)*>

<!ELEMENT pi (#PCDATA |  b | i | u)*>

<!ELEMENT pkg (#PCDATA |  b | i | u)*>

<!ELEMENT b (#PCDATA |  i | u)*>

<!ELEMENT i (#PCDATA |  b | u)*>

<!ELEMENT u (#PCDATA |  b | i)*>
```

Example - RoStorySend

In order to apply roStorySend to implementation of a prompter, you must use this message in conjunction with an roCreate message which sends **all** stories to the prompter from the Running Order (Forced Play List Construction), not just stories which contain the prompter's MOS ID. This establishes a list of all story ID's and pointers in the order they will air for the associated running order. The prompter uses this information to sequence the story information sent in the roStorySend message.

RO messages, such as roCreate, roStoryInsert, roStoryDelete, etc. should be sent before the roStorySend messages. RoStorySend messages can be sent alone after the story is initially referenced in an RO message (e.g. after roCreate or roStoryInsert) if only the body of the story has changed and not it's position within the Running Order.

```
<mos>
   <mosID>prompt.station.com</mosID>
   <ncsID>ncs.station.com</ncsID>
   <roStorySend>
      <roID>96857485</roID>
      <storyID>5983A501:0049B924:8390EF1F</storyID>
      <storySlug>Show Open</storySlug>
      <storyNum>C8</storyNum>
      <mosExternalMetadata>
         <mosScope>PLAYLIST</mosScope>
         <mosSchema>http://NCSA4.com/mos/supported_schemas/NCSAXML2.08</mosSchema>
         <mosPayload>
            <Owner>SHOLMES</Owner>
            <changedBy>MPalmer</changedBy>
            <length>463</length>
            <show>10 pm</show>
         </mosPayload>
      </mosExternalMetadata>
      <storyBody>
         <storyPresenter>Suzie</storyPresenter>
         <storyPresenterRR>10</storyPresenterRR>
         <p>
            <pi> Smile </pi>
         </p>
         <p> Good Evening, I'm Suzie Humpries </p>
         <storyPresenter>Chet </storyPresenter>
```

```
        <storyPresenterRR>12</storyPresenterRR>
        <p> - and I'm Chet Daniels, this is the 5PM news on Monday November 5th.</p>
        <p>First up today - a hotel fire downtown</p>
        <item>
           <itemID>1</itemID>
           <itemSlug>Hotel Fire vo</itemSlug>
           <objID>M000705</objID>
           <mosID>testmos</mosID>
           <itemEdStart>0</itemEdStart>
           <itemEdDur>800</itemEdDur>
           <macroIn>c01/l04/dve07</macroIn>
           <macroOut>r00</macroOut>
           <mosExternalMetadata>
              <mosScope>PLAYLIST</mosScope>
              <mosSchema>http://MOSA4.com/mos/supported_schemas/MOSAXML2.08</mosSchema>
              <mosPayload>
                 <Owner>SHOLMES</Owner>
                 <transitionMode>2</transitionMode>
                 <transitionPoint>463</transitionPoint>
                 <source>a</source>
                 <destination>b</destination>
              </mosPayload>
           </mosExternalMetadata>
        </item>
        <p>...as you can see, the flames were quite high. </p>
     </storyBody>
  </roStorySend>
</mos>
```

5.0 Other messages and data structures

5.1 heartbeat - Connection Confidence Indicator

Purpose

Message sent for the purpose of verifying network and application continuity.

Response

heartbeat

Port

MOS Lower Port (10540) - Media Object Metadata
MOS Upper Port (10541) - Running Order

Structural Outline

mos
- mosID
- ncsID
- heartbeat
  - time

Syntax

```
<!ELEMENT heartbeat (time)>
```

Example

```
<mos>
  <mosID>aircache.newscenter.com</mosID>
  <ncsID>ncs.newscenter.com</ncsID>
  <heartbeat>
    <time>1999-04-11T17:20:42</time>
  </heartbeat>
</mos>
```

5.2 reqMachInfo - Request Machine Information

Purpose

Method for an NCS or MOS to determine more information about it's counterpart.

Response

listMachInfo

Port

MOS Lower Port (10540) - Media Object Metadata
MOS Upper Port (10541) - Running Order

Structural Outline

mos

mosID
ncsID
reqMachInfo

Syntax

```
<!ELEMENT reqMachInfo EMPTY>
```

Example

```
<mos>
  <mosID>aircache.newscenter.com</mosID>
  <ncsID>ncs.newscenter.com</ncsID>
  <reqMachInfo/>
</mos>
```

3.28 listMachInfo - Machine Description List

Purpose

Method for an NCS or MOS to determine more information about it's counterpart.

Response

None

Port

MOS Lower Port (10540) - Media Object Metadata
MOS Upper Port (10541) - Running Order

Structural Outline

mos
  mosID
  ncsID
  listMachInfo
    manufacturer
    model
    hwRev
    swRev
    DOM
    SN
    ID
    time
    opTime?
    mosRev

    mosExternalMetadata*

Syntax

```
<!ELEMENT listMachInfo (manufacturer, model, hwRev, swRev, DOM, SN, ID, time, opTime?,
mosRev, mosExternalMetadata*)>
```

Example

```
<mos>
  <mosID>aircache.newscenter.com</mosID>
  <ncsID>ncs.newscenter.com</ncsID>
  <listMachInfo>
    <manufacturer>RadioVision, Ltd.</manufacturer>
    <model>TCS6000</model>
    <hwRev></hwRev>
    <swRev>2.1.0.37</swRev>
    <DOM></DOM>
    <SN>927748927</SN>
    <ID>airchache.newscenter.com</ID>
    <time>1999-04-11T17:20:42</time>
    <opTime>1999-03-01T23:55:10</opTime>
    <mosRev>2.5</mosRev>
  </listMachInfo>
</mos>
```

5.4 mosExternalMetadata – External Metadata

Purpose

This data block can appear in several messages as a mechanism for transporting additional metadata, independent of schema or DTD.

Behavior

The value of the <scope> tag implies through what production processes this information should travel.

A scope of "OBJECT implies this information is generally descriptive of the object and appropriate for queries.

A scope of "STORY" suggests this information may determine how the Object is used in a Story. For instance, Intellectual Property Management. This information should be stored and used with the Story.

A scope of "PLAYLIST" suggests this information is specific to describing how the Object is to be published, rendered, or played to air and thus, should be included in the Play List in addition to the Story.

This mechanism allows us to roughly filter external metadata and selectively apply it to different production processes and outputs. Specifically, it is neither advisable nor appropriate to send large amounts of inappropriate metadata to the Playlist in roCreate messages. In addition to these blocks of data being potentially very large, the media Object Server is, presumably, already aware of this data.

The value of the <mosSchema> tag should be descriptive of the schema used within the <mosPayload>. The value of <mosSchema> is implied to be a pointer or URL to the actual schema document.

The contents of <mosPayload> must be well formed XML, regardless of the schema used.

Structural Outline

mosExternalMetadata
  mosScope?
  mosSchema
  mosPayload

Syntax

```
<!ELEMENT mosExternalMetadata (mosScope?, mosSchema, mosPayload>

(Note: value of mosSchema is recommended to be a URI - the right most element of which is considered significant and uniquely identifying for the purposes of validation)
```

Example

```
<mosExternalMetadata>
    <mosScope>STORY</mosScope>
     <mosSchema>http://ncsA4.com/mos/supported_schemas/NCSAXML2.08</mosSchema>
    <mosPayload>
      <Owner>SHOLMES</Owner>
      <ModTime>20010308142001</ModTime>
      <mediaTime>0</mediaTime>
      <TextTime>278</TextTime>
      <ModBy>LJOHNSTON</ModBy>
```

```
      <Approved>0</Approved>

      <Creator>SHOLMES</Creator>

</mosPayload>
</mosExternalMetadata>
```

5.5 item – Metadata block transferred by ActiveX Controls included in roCreate messages

Purpose

This data block appears in the MOS Protocol as a subset of the roCreate commands, but may also stand alone as recommended mechanism for transferring Item information from an NCS plug-in to the NCS. It is implied that this format will also be included in the body of the Story and thus will be output in the roStorySend messages.

Behavior

The metadata in the Item Reference is a description of how to execute playback or instantiation of an object, pointed to by the <objID>. The <mosID> is required for forced playlist construction, maintenance of Item References within stories and for association with MOS ActiveX components. The <mosAbstract> field provides a displayable abstract of the Object/Item, more verbose than the <itemSlug>. <mosAbstract> may contain formatting.

It is recommended that vendor or site specific tags be included in the <mosExternalMetadata> structure, not in the body of the message.

Structural Outline

item

- itemID
- itemSlug?
- objID
- mosID
- mosAbstract?
- itemChannel?
- itemEdStart?
- itemEdDur?
- itemTrigger?

macroIn?

macroOut?

mosExternalMetadata*

Syntax

```
<!ELEMENT item (itemID, itemSlug?, objID, mosID, mosAbstract?, itemChannel?,
itemEdStart?, itemEdDur?, itemTrigger?, macroIn?, macroOut?, mosExternalMetadata*)>
```

Example

```
<item>
 <itemID>1</itemID>
 <itemSlug>Man Bites Dog vo</itemSlug>
 <objID>34323</objID>
 <mosID>ncs.com</mosID>
 <mosAbstract>Man Bites Dog vo trt :48</mosAbstract>
 <itemChannel>1</itemChannel>
 <itemEdStart>0</itemEdStart>
 <itemEdDur>1440</itemEdDur>
 <itemTrigger>manual</itemTrigger>
 <macroIn>2e9de</macroIn>
 <macroOut>2399a</macroOut>
 <mosExternalMetadata>
     <mosScope>PLAYLIST</mosScope>
      <mosSchema>http://mosA4.com/mos/supported_schemas/MOSAXML2.08</mosSchema>
      <mosPayload>
       <source>production</source>
       <machine>A5</machine>
     </mosPayload>
 <mosExternalMetadata>
<item>
```

6. Field Descriptions

Many of the terminal elements are constrained in size and/or content as listed below. Character entities count as one character in size constraints. Numeric values may be provided in decimal or hexadecimal (when preceded by "0x", or "x"). Text constants are case sensitive.

b

Bold face type: Specifies that text between tags is in boldface type.

changed

Changed Time/Date: Time the object was last changed in the MOS. Format is YYYY-MM-DD'T'hh:mm:ss[,ddd]['Z'], e.g. 1999-04-11T14:22:07,125Z or 1999-04-11T14:22:07,125-05:00. Parameters displayed within brackets are optional. [,ddd] represents fractional time in which all three digits must be present. ['Z'] indicates time zone which can be expressed as an offset from UTC in hours and minutes. Optionally, the timezone may be replaced by the character 'Z' to indicate UTC.

changedBy

Last Changed by: Name of the person or process that last changed the object in the MOS. This can be stored in a language other than English.

command

roItemCtrl command: The commands READY, EXECUTE, PAUSE and STOP, as well as general indicator, SIGNAL, can be addressed at each MOS Structure level. In other words, a single command can begin EXECUTION of an entire Running Order, of a Story containing multiple Items, or of a single Item.

created

Creation Time/Date: Time the object was created in the MOS. Format is YYYY-MM-DD'T'hh:mm:ss[,ddd]['Z'], e.g. 1999-04-11T14:22:07,125Z or 1999-04-11T14:22:07,125-05:00. Parameters displayed within brackets are optional. [,ddd] represents fractional time in which all three digits must be present. ['Z'] indicates time zone which can be expressed as an offset from UTC in hours and minutes. Optionally, the timezone may be replaced by the character 'Z' to indicate UTC.

createdBy

Created by: Name of the person or process that created the object in the MOS. This can be stored in a language other than English. 128 chars max.

description

Object Description: Text description of the MOS object. No maximum Length is defined. This can be stored in a language other than English.

DOM

Date of Manufacture.

em

Emphasized Text: markup within description and p to emphasize text.

hwRev

HW Revision: 128 chars max.

I

Italics: Specifies that text between tags is in Italics.

ID

Identification of a Machine: text. 128 chars max.

item

Item: Container for item information within a Running Order message.

itemChannel

Item Channel: Channel requested by the NCS for MOS to playback a running order item. 128 chars max.

itemEdDur

Item Editorial Duration: in number of samples 0XFFFFFFFF max.

itemEdStart

Editorial Start: in number of samples 0XFFFFFFFF max.

itemID

Item ID: Defined by NCS, UID not required. 128 chars max.

itemSlug

Item Slug: Defined by NCS. 128 chars max

itemTrigger

Item Air Trigger: “MANUAL”, “TIMED” or “CHAINED”.

CHAINED (sign +/-) (value in # of samples)

CHAINED -10 would start the specified clip 10 samples before the proceeding clip ended.

CHAINED 10 would start the specified clip 10 samples after the preceding clip ended, thus making a pause of 10 samples between the clips. There is a space character between the word CHAINED and the value.

macroIn

Macro Transition In: Defined by MOS. 128 chars max.

macroOut

Macro Transition Out: Defined by MOS. 128 chars max.

manufacturer

Manufacturer: Text description. 128 chars max.

model

Model: Text description. 128 chars max.

modifiedBy

Modified by: Name of the person or process that last modified the object in the MOS. This can be stored in a language other than English. 128 chars max.

mosAbstract

Abstract of the Object intended for display by the NCS. This field may contain HTML and DHTML markup. The specific contents are limited by the NCS vendor's implementation. Length is unlimited but reasonable use is suggested.

mosID

MOS ID: Character name for the MOS unique within a particular installation.

mosGroup

This field is intended to imply the name of a destination, group or folder for the Object pointer to be stored in the NCS. 128 chars max.

mosRev

MOS Revision: Text description. 128 chars max.

mosScope

This field implies the extent to which the mosExternalMetadata block will move through the NCS workflow. Accepted values are "OBJECT" "STORY" and "PLAYLIST"

ncsID

NCS ID: Character name for the NCS unique within a particular installation. 128 chars max.

objAir

Air Status: “READY” or “NOT READY”.

objDur

Object Duration: The number of samples contained in the object. For Still Stores this would be 1. 0XFFFFFFFF MAX

objID

Object UID: Unique ID generated by the MOS and assigned to this object. 128 chars max.

objRev

Object Revision Number: 999 max.

objSlug

Object Slug: Textual object description. 128 chars max.

objTB

Object Time Base: Describes the sampling rate of the object in samples per second. For still stores this is 0. For PAL Video this would be 50. For NTSC it would be 60. For audio it would reflect the audio sampling rate. Object Time Base is used by the NCS to derive duration and other timing information. 0XFFFFFFFF MAX

objType

Object Type: Choices are “STILL”, “AUDIO”, “VIDEO”.

opTime

Operational Time: date and time of last machine start. Format is YYYY-MM-DD'T'hh:mm:ss[,ddd]['Z'], e.g. 1999-04-11T14:22:07,125Z or 1999-04-11T14:22:07,125-05:00. Parameters displayed within brackets are optional. [,ddd] represents fractional time in which all three digits must be present. [‘Z’] indicates time zone which can be expressed as an offset from UTC in hours and minutes. Optionally, the timezone may be replaced by the character ‘Z’ to indicate UTC.

p

Paragraph: Standard html delimitation for a new paragraph.

pause

Item Delay: Requested delay between items in ms 0XFFFFFFFF MAX.

pi

Presenter instructions: Instructions to the anchor or presenter that are not to be read such as

"Turn to 2-shot."

pkg

Package: Specifies that text is verbatim package copy as opposed to copy to be read by presenter.

roAir

Air Ready Flag: "READY" or "NOT READY".

roChannel

Running Order Channel: default channel requested by the NCS for MOS to playback a running order. 128 chars max.

roCtrlCmd

Running Order Control Command: READY, EXECUTE, PAUSE and STOP, as well as general indicator, SIGNAL, can be addressed at each level. In other words, a single command can begin EXECUTION of an entire Running Order, of a Story containing multiple Items, or of a single Item.

roCtrlTime

Running Order Control Time: roCtrlTime is an optional field which provides a mechanism to time stamp the time of message transmission, or optionally, to provide a time in the immediate future at which the MOS should execute the command. Format is YYYY-MM-DD'T'hh:mm:ss[,ddd]['Z'], e.g. 1999-04-11T14:22:07,125Z or 1999-04-11T14:22:07,125-05:00. Parameters displayed within brackets are optional. [,ddd] represents fractional time in which all three digits must be present. ['Z'] indicates time zone which can be expressed as an offset from UTC in hours and minutes. Optionally, the timezone may be replaced by the character 'Z' to indicate UTC.

roEdDur

Running Order Editorial Duration: duration of entire running order. Format in hh:mm:ss, e.g. 00:58:25.

roEdStart

Running Order Editorial Start: date and time requested by NCS for MOS to start playback of a running order. Format is YYYY-MM-DD'T'hh:mm:ss[,ddd]['Z'], e.g. 1999-04-11T14:22:07,125Z or 1999-04-11T14:22:07,125-05:00. Parameters displayed within brackets are optional. [,ddd] represents fractional time in which all three digits must be present. ['Z'] indicates time zone which can be expressed as an offset from UTC in hours and minutes. Optionally, the timezone may be replaced by the character 'Z' to indicate UTC.

roEventTime

Running Order Event Time: Time of the time cue sent to the parent MOS by the NCS for a specific item event.

roEventType

Running Order Event Type: The type of event that is being queued in the Running order.

roID

Running Order UID: Unique Identifier defined by NCS. 128 chars max.

roSlug

Running Order Slug: Textual Running Order description. 128 chars max.

roStatus

Running Order Status: Options are: "OK" or error description. 128 chars max.

roTrigger

Running Order Air Trigger: “MANUAL”, “TIMED” or “CHAINED”.

CHAINED (sign +/-) (value in # of samples)

CHAINED -10 would start the specified clip 10 samples before the proceeding clip ended. CHAINED 10 would start the specified clip 10 samples after the preceding clip ended, thus making a pause of 10 samples between the clips. There is a space character between the word CHAINED and the value.slug Textual Object ID: This is the text slug of the object and is stored in the native language. This can be stored in a language other than English. 128 chars max.

SN

Serial Number: text serial number. 128 chars max.

status

Status: Options are “NEW” “UPDATED” “MOVED” “BUSY “ “DELETED”, "NCS CTRL", "MANUAL CTRL", "READY", "NOT READY", "PLAY," "STOP".

statusDescription

Status Description: textual description of status. 128 chars max.

story

Story: Container for story information in a Running Order message.

storyBody

Story Body: The actual text of the story within a running order.

storyID

Story UID: Defined by the NCS. 128 chars max.

storyItem

Story Item: An item imbedded into a story that can be triggered when that point in the story is reached in the teleprompter.

storyNum

Story Number: The name or number of the Story as used in the NCS. This is an optional field originally intended for use by prompters. 128 chars max.

storyPresenter

Story Presenter: The anchor or presenter of a story within an running order.

storyPresenterRR

Story Presenter Read Rate: The read rate of the anchor or presenter of a story within a running order.

storySlug

Story Slug: Textual Story description. 128 chars max.

swRev

Software Revision: (MOS) Text description. 128 chars max.

tab

Tab: tabulation markup within description and p.

time

Time: Time object changed status. Format is YYYY-MM-DD'T'hh:mm:ss[,ddd]['Z'], e.g. 1999-04-11T14:22:07,125Z or 1999-04-11T14:22:07,125-05:00. Parameters displayed within brackets are optional. [,ddd] represents fractional time in which all three digits must be present. ['Z'] indicates time zone which can be expressed as an offset from UTC in hours and minutes. Optionally, the timezone may be replaced by the character 'Z' to indicate UTC.

u

Underline: Specifies that text between tags is to be underlined.

7. Recent Changes

7.0 Changes to MOS version 2.6 WD-2001-08-09

1. Removed second reference to mosExternalMetadata in roStorySend description
2. DTD reworked with suggestions from Jiri Basek Aveco s.r.o (Jiri.Basek@aveco.com)
3. DTD also posted to web site at http://www.mosprotocol.com/mos2dot601.dtd

7.1 Changes from MOS version 2.5 to 2.6 WD-2001-06-06

1. Added message for mosItemReplace.
2. Added message for roMetadataReplace.
3. Added message for roStoryMove.
4. Added structure for mosExternalMetadata.
5. mosExternalMetadata structure added to many common elements
6. DTD was restructured to include new and changed elements. The sequence of definitions was also changed.

7.2 Changes from MOS version 2.0 to 2.5 WD-2000-08-01

7. Added tag for mosObjCreate.
8. Added tag for roStorySend.
9. Added tag for roItemCue.
10. Added tag for roCtrl.
11. Clarified Definition of roList and *roReq*
12. Fixed errors and syntax errors in documentation.
13. Modified Item structure. Description is as follows:

Modified Item Structure

Purpose

Allows for identification of the parent Media Object Server when the play list is forcefully constructed on a machine other than an object's parent Media Object Server, i.e. when a play list is constructed in a MOS Prompter.

Description

The format of the message remains the same. The only change is the optional addition of the existing mosID tag. The value of this tag, if appropriately used, is set to the ID of the referenced object's parent Media Object Serve. This change is reflected throughout all MOS messages which include the <item> structure.

Structural Outline

item
- itemID
- itemSlug
- mosID
- objID
- itemChannel
- itemEdStart
- itemEdDur

itemTrigger
macroIn
macroOut

7.3 Changes for MOS 2.0 WD-1999-05-12

1. Restructured Running Order messages so that running orders contain "stories", which contain "items".
2. Added optional channel assignment tag "itemChannel" following objID in item tag in Running Order messages.
3. Added story tag "story" to Running Order messages to contain storyID and Items.
4. Changed the name of message types so that messages that are exclusive to the Media Object Metadata socket start with "mos" and messages exclusive to the Running Order socket start with "ro".
5. Eliminated Running Order item messages. The same effect can be achieved using Running Order story messages.
6. Changed item slug tag from "slug" to "itemSlug" and added optional "storySlug" tag following story tag.
7. Added message "mosReqObj" to allow NCS to request MOS object information for a specific object by objID. Response is "mosObj".
8. Added optional tags for Running Order channel, start, duration and trigger (roChannel, roEdStart, roEdDur, roTrigger). Renamed item tags (itemChannel, itemEdStart, itemEdDur, itemTrigger).
9. Added message "roReqAll" for MOS to request list of all known running orders from an NCS. Response is "roListAll" message.
10. Changed "metadata" tag to "statusDescription" in mosAck and to "description" in mosObj message.
11. Added itemID to "roItemStat" message.

7.4 Changes from MOS version 1.52 to 2.0 WD-1999-03-17

1. Changed tags to XML format.
2. Added root tag "mos".
3. Removed space in tags.
4. Change MOS ID and NCS ID to always be in the same sequence.
5. Tags must be used in the sequence defined.
6. Changed capitalization of tags for readability.
7. Added optional formatting to metadata.

8. MOS 2.601 DTD

<!-- MOS.DTD version 2.601 August 9, 2001-->

<!-- edited with XML Spy v4.0 NT beta 2 build Jul 26 2001 (http://www.xmlspy.com) by Mike Palmer (Associated Press) -->

<!--Thanks to Jiri Basek Aveco s.r.o (Jiri.Basek@aveco.com) and the Octopus Team for the following list of modifications due to original DTD bugs :

1: element "roStorySend" : comma added between "mosExternalMetadata*" and "storyBody"

2: element "roStorySend" old version : deactivated

3: element "item" : ending bracket added after "mosExternalMetadata*"

4: element "command' : "READY", "EXECUTE", "PAUSE", "STOP", "SIGNAL" are not elements but predefined strings

5: element "mosObj" : 'externalData" changed to "mosExternalData'

6: ATTLIST metadata : deactivated. ? should it be changed to ATTLIST mosExternalMetadata ?

8: element "mosScope" : "object, "story", "playlist" are not elements but predefined strings

A9: element "mos" : "reqMachineInfo" changed to "reqMachInfo"

A10: element "storyNum" defined

Additional changes have been made to the formatting and sequence of the elements in order to better match the order of structures presented in the MOS Protocol document.

-->

<!-- MOS Message - One message type per message -->

<!ELEMENT mos (mosID, ncsID, (mosAck | mosObj | mosReqObj | mosReqAll | mosListAll | mosObjCreate | mosItemReplace | roAck | roCreate | roReplace | roMetadataReplace | roDelete | roReq | roList | roReqAll | roListAll | roStat | roReadyToAir | roStoryAppend | roStoryInsert | roStoryReplace | roStoryMove | roStorySwap | roStoryDelete | roItemStat | roItemCue | roCtrl | roStorySend | heartbeat | reqMachInfo | listMachInfo))>

<!ELEMENT mosAck (objID, objRev, status, statusDescription)>

<!ELEMENT mosObj (objID, objSlug, mosAbstract, objGroup?, objType, objTB, objRev, objDur, status, objAir, createdBy, created, changedBy, changed, description, mosExternalMetadata*)>

<!ELEMENT mosReqObj (objID)>

<!ELEMENT mosReqAll (pause)>

<!ELEMENT mosListAll ((objID, objSlug, mosAbstract?, objGroup?, objType, objTB, objRev, objDur, status, objAir, createdBy, created, changedBy, changed, description, mosExternalMetadata*)*)>

<!ELEMENT mosObjCreate (objSlug, objGroup?, objType, objTB, objDur?, time?, createdBy?, description?, mosExternalMetadata*)>

<!ELEMENT mosItemReplace (roID, storyID, item)>

<!ELEMENT roAck (roID, roStatus, (storyID, itemID, objID, status)*)>

<!ELEMENT roCreate (roID, roSlug, roChannel?, roEdStart?, roEdDur?, roTrigger?, macroIn?, macroOut?, mosExternalMetadata*, story*)>

<!ELEMENT roReplace (roID, roSlug, roChannel?, roEdStart?, roEdDur?, roTrigger?, macroIn?, macroOut?, mosExternalMetadata*, story*)>

<!ELEMENT roMetadataReplace (roID, roSlug, roChannel?, roEdStart?, roEdDur?, roTrigger?, mosExternalMetadata?)>

<!ELEMENT roDelete (roID)>

<!ELEMENT roReq (roID)>

<!ELEMENT roList (roID, roSlug, roChannel?, roEdStart?, roEdDur?, roTrigger?, macroIn?, macroOut?, mosExternalMetadata*, story*)>

<!ELEMENT roListAll ((roID, roSlug?, roChannel?, roEdStart?, roEdDur?, roTrigger?, mosExternalMetadata*)*)>

<!ELEMENT roStat (roID, status, time)>

<!ELEMENT roReadyToAir (roID, roAir)>

<!ELEMENT roStoryAppend (roID, story+)>

<!ELEMENT roStoryInsert (roID, storyID, story+)>

<!ELEMENT roStoryReplace (roID, storyID, story+)>

<!ELEMENT roStoryMove (roID, storyID, storyID)>

<!ELEMENT roStorySwap (roID, storyID, storyID)>

<!ELEMENT roStoryDelete (roID, storyID+)>

<!ELEMENT roStorySend (roID, storyID, storySlug?, storyNum?, storyBody, mosExternalMetadata*)>

```
<!ELEMENT roItemStat (roID, storyID, itemID, objID, status, time)>

<!ELEMENT roItemCue (mosID, roID, storyID, itemID, roEventType, roEventTime, mosExternalMetadata*)>

<!ELEMENT roCtrl (roID, storyID, itemID, command, mosExternalMetadata*)>

<!ELEMENT heartbeat (time)>

<!ELEMENT listMachInfo (manufacturer, model, hwRev, swRev, DOM, SN, ID, time, opTime?, mosRev, mosExternalMetadata*)>

<!ELEMENT story (storyID, storySlug?, storyNum?, mosExternalMetadata*, item*)>

<!ELEMENT description (#PCDATA | p | em | tab)*>

<!ELEMENT storyBody (storyPresenter*, storyPresenterRR*, p*, storyItem*)>

<!ELEMENT storyItem (itemID, itemSlug?, objID, mosID, mosAbstract?, itemChannel?, itemEdStart?, itemEdDur?, itemTrigger?, macroIn?, macroOut?,
mosExternalMetadata*)>

<!ELEMENT item (itemID, itemSlug?, objID, mosID, mosAbstract?, itemChannel?, itemEdStart?, itemEdDur?, itemTrigger?, macroIn?, macroOut?,
mosExternalMetadata*)>

<!ELEMENT mosExternalMetadata (mosScope?, mosSchema, mosPayload)>

<!ELEMENT b (#PCDATA | i | u)*>

<!ELEMENT i (#PCDATA | b | u)*>

<!ELEMENT p (#PCDATA | em | tab | pi | pkg | b | i | u)*>

<!ELEMENT pi (#PCDATA | b | i | u)*>

<!ELEMENT pkg (#PCDATA | b | i | u)*>

<!ELEMENT u (#PCDATA | b | i)*>

<!ELEMENT mosID (#PCDATA)>

<!ELEMENT ncsID (#PCDATA)>

<!ELEMENT changed (#PCDATA)>

<!ELEMENT changedBy (#PCDATA)>

<!ELEMENT command (#PCDATA)>

<!-- valid values for "command" are "READY", "EXECUTE", 'PAUSE', "STOP", "SIGNAL" -->

<!ELEMENT created (#PCDATA)>

<!ELEMENT createdBy (#PCDATA)>

<!ELEMENT DOM (#PCDATA)>

<!ELEMENT em (#PCDATA)>

<!ELEMENT hwRev (#PCDATA)>

<!ELEMENT ID (#PCDATA)>

<!ELEMENT itemChannel (#PCDATA)>

<!ELEMENT itemEdDur (#PCDATA)>

<!ELEMENT itemEdStart (#PCDATA)>

<!ELEMENT itemID (#PCDATA)>

<!ELEMENT itemSlug (#PCDATA)>

<!ELEMENT itemTrigger (#PCDATA)>

<!ELEMENT macroIn (#PCDATA)>

<!ELEMENT macroOut (#PCDATA)>

<!ELEMENT manufacturer (#PCDATA)>

<!ELEMENT model (#PCDATA)>
```

```
<!ELEMENT mosAbstract ANY>
<!ELEMENT mosPayload ANY>
<!ELEMENT mosSchema (#PCDATA)>
<!ELEMENT mosScope (#PCDATA)>
<!-- valid values for "mosScope" are "OBJECT", "STORY", "PLAYLIST" -->
<!ELEMENT mosRev (#PCDATA)>
<!ELEMENT objAir (#PCDATA)>
<!ELEMENT objDur (#PCDATA)>
<!ELEMENT objGroup (#PCDATA)>
<!ELEMENT objID (#PCDATA)>
<!ELEMENT objRev (#PCDATA)>
<!ELEMENT objSlug (#PCDATA)>
<!ELEMENT objTB (#PCDATA)>
<!ELEMENT objType (#PCDATA)>
<!ELEMENT opTime (#PCDATA)>
<!ELEMENT pause (#PCDATA)>
<!ELEMENT reqMachInfo EMPTY>
<!ELEMENT roAir (#PCDATA)>
<!ELEMENT roID (#PCDATA)>
<!ELEMENT roChannel (#PCDATA)>
<!ELEMENT roCtrlCmd (#PCDATA)>
<!ELEMENT roCtrlTime (#PCDATA)>
<!ELEMENT roEdDur (#PCDATA)>
<!ELEMENT roEdStart (#PCDATA)>
<!ELEMENT roEventTime (#PCDATA)>
<!ELEMENT roEventType (#PCDATA)>
<!ELEMENT roReqAll EMPTY>
<!ELEMENT roSlug (#PCDATA)>
<!ELEMENT roStatus (#PCDATA)>
<!ELEMENT roTrigger (#PCDATA)>
<!ELEMENT SN (#PCDATA)>
<!ELEMENT status (#PCDATA)>
<!ELEMENT statusDescription (#PCDATA)>
<!ELEMENT storyID (#PCDATA)>
<!ELEMENT storyNum (#PCDATA)>
<!ELEMENT storyPresenter (#PCDATA)>
<!ELEMENT storyPresenterRR (#PCDATA)>
<!ELEMENT storySlug (#PCDATA)>
<!ELEMENT swRev (#PCDATA)>
<!ELEMENT tab (#PCDATA)>
```

```
<!ELEMENT time (#PCDATA)>
<!-- Attributes -->
<!ATTLIST mos
    version CDATA #FIXED "-//MOS Group//DTD MOS 2.601//EN"
    changeDate CDATA #FIXED "09 August 2001"
>
<!-- <!ATTLIST metadata xml:space (default | preserve) 'preserve'>  -->
```

8. References and Resources

8.1

The primary site for MOS protocol information is http://www.mosprotocol.com.

8.2 XML FAQ

http://www.ucc.ie/xml/ contains an extensive document of Frequently Asked Questions regarding XML.

8.3 Recommended Reading

Just XML: John E. Simpson; 1999. Prentice Hall PTR. ISBN 0-13-9434417-8. ($34.99)
XML for Dummies Quick Reference: Mariva H. Aviram; 1998. IDG Books Worldwide, Inc. ISBN 0-7645-0383-9. ($14.99)

The Unicode Standard, Version 2.0: The Unicode Consortium. Addison-Wesley. ISBN 0-201-48345-9. ($62.95)

8.4 XML Web Sites

The mission of XML.com is to help you discover XML and learn how this new Internet technology can solve real-world problems in information management and electronic commerce.
http://www.xml.com/xml/pub/

Robin Cover's XML resource page is perhaps the most useful and extensive available on the Web.
http://www.oasis-open.org/cover/xml.html

APPENDIX B

GLOSSARY

OSM Definitions and Functional Descriptions

Modules

MOS Protocol Receiver

Receives Unicode XML text message via TCP/IP Socket and saves each message as a file. This module also handles message acknowledgment and transmission of status messages passed to the module.

MOS File Parser

Validates each XML message and parses appropriate messages for mosObj pointers embedded in Item References.

XML & Object File Storage

Stores XML files and uses mosObj pointers to retrieve referenced media objects which it also stores.

Business Rules

Business rules are applied to metadata included in the RO, Story, Item and Object levels. These rules primarily control routing of content through the system. Business rules are also applied to the current status of transmission modules, level of content storage used in the OSM and in remote clients.

Addressing Module

The addressing module takes plain language feed and client names and resolves these to delivery destinations which may include IP addresses, multi-cast addresses, or file folder destinations.

File Transmit Module

The file transmit module takes addressing information and uses this to deliver XML and Object files to the proper storage location and in the proper format for the chosen delivery mechanism to recognize and forward them to the client destinations.

Internet FTP Interface

The FTP interface uses control information sent to it from the File Transmit Module to determine what files should be sent to remove clients over the Internet or network. This module will scale from handling one or two FTP sessions to multiple FTP sessions in the first release.

This interface is also capable of passing information upstream back to the OSM. This information should include H&S (Health and Status) as well as the means for support of e-commerce models.

The interface is able to transmit multiple files simultaneously to individual clients, in addition to supporting simultaneous transmission to multiple clients.

Four possible channels are provided:

1) roConstruction
2) roStorySend
3) low res proxies
4) high res content

SkyStream Interface

The SkyStream interface uses control information sent to it from the File Transmit module to determine what files should be sent to remove clients over a satellite transmission path. Other control information is also sent from the File Transmit Module to the SkyStream interface which may control the level of forward error correction applied, the frequency of repetition and priority of files in the queue.

This interface is also capable of passing information upstream back to the OSM. This information should include H&S (Health and Status) as well as the means for support of e-commerce models.

The interface is able to transmit multiple files simultaneously to individual clients, in addition to supporting simultaneous transmission to multiple clients.

Four possible channels are provided:

1) roConstruction
2) roStorySend
3) low res proxies
4) high res content

OSM Client

Modules

Internet FTP Interface

Works as a client to the OSM Internet FTP interface.

SkyStream Interface

Works as a client to the OSM SkyStream interface.

File Receive Module

Takes files from the SkyStream and FTP interfaces and stores them to disk.

MOS File Parser

This is the same MOS File Parser as used in the OSM. Validates each XML message and parses appropriate messages for mosObj pointers embedded in Item References

XML & Object File Storage

Nominally 80 gB or more of storage for both media objects and XML files.

Business Rule Module

Nominally, the same engine as used in the OSM or a derivative. Business Rules will include purge of media objects no longer referenced in Stories or Running Orders, holding of embargoed material, compilation of usage statistics, and transmission of statistics and H&S via transmission interfaces.

ASP Web Server

The server provides functionality as defined for the Browser UI, specified below. It must also handle streaming of the low resolution proxy files.

This also includes an FTP interface which exposes XML and Object files for other devices to retrieve.

Video Output

As commanded by the ASP interface, the Video Output plays out the full resolution version of the requested object. If the requested object does not exist or has not completed transmission, the Video Output module may alternately output the low resolution proxy, scaled to full screen if it exists.

This module is capable of queuing multiple requests and "slating" each playback with 5 seconds of title and/or other information extracted from Story and Object metadata.

Optionally, the module is capable of basic Sony Beta Serial Control to cue legacy tape machines to record and stop.

MOS Output

MOS messages may optionally be forwarded, as appropriate and as they conform to business rules, to a local NCS where they can be used to recreate the RO, Stories, and Object Pointers.

Serial Wire Output

XML Story information can optionally be reformatted via XSL transforms into a number of different agency wire transmission formats. These text files are then transmitted via a serial interface at a configurable speed. No flow control or feedback is required.

Web Browser ASP User Interface

Content Explorer

The Content Explorer is comprised of some number of ASP pages which present:

1) A list of Running Orders
2) A list of Stories in a selected Running Order
3) The text of a selected Story
4) A list of Media Objects in a selected Story
5) A proxy version of a selected media object

The Interface allows the user to do general searches on both text and metadata contained in media object pointers. The interface must allow remote control play-out of video from the server from the Web Browser UI. The interface also includes e-commerce functions. The interface is updateable via the same transmission mechanisms as are used to transmit XML and Object files.

United States Patent & Trademark Office

Office of Initial Patent Examination -- Scanning Division